601.433/633 Introduction to Algorithms **Fall 2024**
Homework #2 **Due:** September 12, 2024, 9am

Remember: you may work in groups of up to three people, but must write up your solution entirely on your own. Collaboration is limited to discussing the problems – you may not look at, compare, reuse, etc. any text from anyone else in the class. Please include your list of collaborators on the first page of your submission. You may use the internet to look up formulas, definitions, etc., but may not simply look up the answers online.

Please include proofs with all of your answers, unless stated otherwise.

1 Reversals (33 pts)

Given an array $[a_1, a_2, \ldots, a_n]$, a reversal is a pair (i, j) such that $i < j$ but $a_i > a_j$. For example, in the array $[5, 3, 2, 10]$ there are are three reversals $((1, 2), (1, 3), (2, 3))$. Note that the array has no reversals if and only if it is sorted, so the number of reversals can be thought of as a measure of how well-sorted an array is.

(a) (11 points) What is the expected number of reversals in a random array? More formally, consider a uniformly random permutation of n distinct elements $a_1, \ldots, a_n$: what is the expected number of reversals? Give your answer precisely, without asymptotic notation. Prove your answer.

(b) (11 points) Recall the insertion sort algorithm:

```
for i = 1 to n
    j = i
    while j > 0 and A[j-1] > A[j]
        swap A[j] and A[j-1]
        j = j - 1
```

Suppose that our array has d reversals. Prove that the (worst-case) running time of insertion sort is $\Theta(n + d)$.

(c) (11 points) What does this imply about the *average-case* running time of insertion sort as a function only of n? That is, if we draw a permutation uniformly at random, what is the expected running time of insertion sort (in asymptotic notation)? Note that this is *not* a randomized algorithm; this is a deterministic algorithm on a random input.

2 Pitcher Matching (34 points)

Suppose that you are given n red and n blue water pitchers, all of different shapes and sizes. All red pitchers hold different amounts of water, as do all the blue ones. Moreover, for every red pitcher there is a blue pitcher that holds exactly the same amount of water (and vice versa).

Your job is to find a matching between red pitchers and blue pitchers that hold the same amount of water. In other words, you want to identify for every red pitcher which blue pitcher has the same volume, and similarly for every blue pitcher which red pitcher has the same volume. To do this, you are only allowed to use the following operation: pick a red pitcher and a blue pitcher, fill the red pitcher, and pour it into the blue pitcher. This will tell you whether the volume of the red pitcher is less than, equal to, or greater than the volume of the blue pitcher. In other words, you can compare any red pitcher and any blue pitcher. But you *cannot* compare two red pitchers or two blue pitchers.

(a) (16 points) Give a randomized algorithm that uses $O(n \log n)$ comparisons in expectation.

(b) (9 points) Prove that your algorithm is correct (i.e., that it always returns the correct matching).

(c) (9 points) Prove that your algorithm uses $O(n \log n)$ comparisons in expectation.

3 Searching an Array Randomly (33 points)

Let A be an unsorted array of length n, where each entry of A is an integer. Suppose that we are looking for some integer x in A, i.e., we want to find an index i such that $A[i] = x$ if such an index exists. If no such index exists, we should return False. Consider the following randomized algorithm.

- Initially, all indices are unmarked.
- While not all indices are marked:
  - Pick an index $i \in [n]$ uniformly at random.
  - If $A[i] = x$ return i.
  - Else mark index i.
- Return false

Note that in each iteration we pick an index i uniformly at random from $[n]$, not from the set of unmarked indices. So we might examine a given index more than once.

(a) (16 points) Suppose that x appears in $k \geq 1$ places in A, i.e., $|\{i : A[i] = x\}| = k$. What is the expected running time of this algorithm, as a function of n and k?

Hint: if you are unfamiliar with Bernoulli and Geometric distributions, consider reading CLRS (3rd edition) Appendix C.4.

(b) (17 points) Suppose that x does not appear in A. What is the expected running time of the algorithm?